Non-linear Autoregressive Neural Networks to Forecast Short-Term Solar Radiation for Photovoltaic Energy Predictions

Non-linear Autoregressive Neural Networks to forecast Short-term Solar Radiation for Photovoltaic Energy Predictions*

Alessandro Aliberti[1][0000−0001−8828−608X], Lorenzo Bottaccioli[1][0000−0001−7445−3975], Giansalvo Cirrincione[3][0000−0002−2894−4164], Enrico Macii[2][0000−0001−9046−5618], Andrea Acquaviva[2][0000−0002−7323−759X], and Edoardo Patti[1][0000−0002−6043−6477]

[1] Dept. of Control and Computer Engineering, Politecnico di Torino, Italy
[2] Interuniversity Dept. of Regional and Urban Studies and Planning, Politecnico di Torino, Italy
[3] Universite de Picardie Jules Verne, Amiens, France
{name,surname}@polito.it, giansalvo.cirrincione@u-picardie.fr

Abstract. Nowadays, green energy is considered as a viable solution to hinder CO_2 emissions and greenhouse effects. Indeed, it is expected that Renewable Energy Sources (RES) will cover 40% of the total energy request by 2040. This will move forward decentralized and cooperative power distribution systems also called smart grids. Among RES, solar energy will play a crucial role. However, reliable models and tools are needed to forecast and estimate with a good accuracy the renewable energy production in short-term time periods. These tools will unlock new services for smart grid management.

In this paper, we propose an innovative methodology for implementing two different non-linear autoregressive neural networks to forecast Global Horizontal Solar Irradiance (GHI) in short-term time periods (i.e. from future 15 to 120 min). Both neural networks have been implemented, trained and validated exploiting a dataset consisting of four years of solar radiation values collected by a real weather station. We also present the experimental results discussing and comparing the accuracy of both neural networks. Then, the resulting GHI forecast is given as input to a Photovoltaic simulator to predict energy production in short-term time periods. Finally, we present the results of this Photovoltaic energy estimation discussing also their accuracy.

Keywords: Solar radiation forecast · Artificial neural networks · AR · ARMA · Dynamic system · Photovoltaic system · Energy forecast · Renewable energy.

* This work was partially supported by the Italian project "Edifici a Zero Consumo Energetico in Distretti Urbani Intelligenti".

1 Introduction

Nowadays, to contrast negative effects of pollution, global warming and waste of energy, green energy represents a very attractive solution, especially solar energy [13]. Indeed, applications like Photovoltaic (PV) systems are changing the electrical energy production, consumption and distribution in our cities [21]. We are witnessing the transaction of our society from centralized and hierarchical power distribution systems to distributed and cooperative ones, generally called Smart Grids. The technology introduced by this new philosophy is opening the electrical marketplace to new actors (e.g. prosumers and energy aggregators). In classic power grids, the stability is achieved by consolidated generation plants using primary and secondary reserve at large-scale [14]. Whilst, in a Smart Grid scenario, new actors can actively contribute to load-balancing by fostering novel services for network management and stability. Demand/Response [40] is an example of such applications for Smart Grid management. It permits to achieve a temporary virtual power plant [45] by changing the energy consumption patterns of consumers i) to match energy produced by renewable energy systems or ii) to fulfil grid operation requirements. This process is generally done every 15 minutes. In these applications, the amount of available energy must be known in advance to optimize the production of power plants [1] and to match energy production with consumption. Thus, we need tools to forecast with a good accuracy of solar radiation and, consequently, solar energy.

Several studies were proposed in the literature to find mathematical and physical models to estimate and forecast solar radiation, such as stochastic models based on time-series [22], [46], [3]. Moreover, classical linear time-series models have been widely used [8]. However, these studies have proven that these methodologies often are not sufficient in the analysis and prediction of solar radiation. This is due to the non-stationary and non-linearity of solar radiation time-series data [26], [31]. Furthermore, stochastic models are based on the probability estimation. This leads to a difficult forecast of the solar radiation time-series. To overcome these limits, non-linear approaches, such as Artificial Neural Networks (ANNs), were considered by many researchers as powerful tools to predict such phenomenons [47]. Generally, ANNs do not require knowledge of internal system parameters and they offer a compact solution for multiple variable problems [35]. However, also the use of an ANN to forecast a phenomenon introduces an error, the so-called *prediction error* [53]. As a result, these models need optimizations to reduce this error. With respect to presented literature solutions to forecast solar radiation, the scientific novelty of our methodology consists of using Multilayer Perceptron, which is the artificial neural network most used for this kind of applications [12]. Generally, most literature methodologies rely on the single past value to perform the forecast [7]. Whilst, the proposed solutions allow to reduce significantly the *prediction error* by using a set of regressors to perform predictions, as discussed in our previous work [2]. However, differently from [2], our goal is to better optimize the neural architecture by adding further levels of difficulty. In particular, we used a more complex and larger dataset to better

forecast the solar radiation in short-term, i.e. from future 15 minutes up to next 2 hours.

In this paper, we present and compare two Nonlinear Autoregressive neural networks to forecast short-term solar radiation that is then applied to estimate PV energy production. We trained and validated both neural networks with a dataset consisting of four years of Global Horizontal Solar Irradiance (GHI) samples collected by a real weather station. Both neural networks are Multilayer Perceptron based and they exploit a certain number of regressors to predict GHI in a range of next 15 minutes up to 2 hours. Then, GHI forecast is given as input to our PV simulator [6] that exploits GIS (Geographic Information System) tools to simulate energy production. We also provide an exhaustive comparison of this work with our previous paper [2], highlighting differences and improvements. Finally, we discuss advantages and disadvantages of choosing a neural network rather than another, among those developed and analyzed.

The rest of the paper is organized as follows. Section 2 discusses the followed methodology to define our neural networks for short-term solar radiation forecast. Section 3 details all the steps performed to initialize, train and validate both our neural networks. Section 4 presents the results of solar radiation forecast given by the proposed ANNs. Section 5 briefly introduce our PV simulator [6] and, then, presents the estimation results on PV energy simulations that exploit foretasted solar radiation output of the proposed neural networks. Finally, Section 6 discusses our concluding remarks.

2 Methodology

Predicting the energy producuction of a PV system means being able to forecast the level of GHI to which the PV system is exposed to. In turn, predicting the values of GHI means working with time-series information. This kind of information identifies a sequence of values chronologically ordered [16]. The study and manipulation of time-series models brings different benefits. Mainly, it allows: i) in understanding the underlying forces and structures that produced the observed data and ii) in fiting a model and in proceeding to forecast and monitor or even feedback and feed-forward control [34].

2.1 The Multilayer Perceptron

Nowadays, one of the most effective methods for prediction is based on neural networks [30]. This is due to their versatility and their ability to model a wide range of systems reducing development time and offering better performances [49]. In the study of systems based on time-series, the most powerful and performing family of ANNs is the Multilayer Perceptron (MLP) [12]. These types of neural networks are composed of units, called nodes or neurons, and organized in a layer of inputs, one or more hidden layers and an output layer. The MLP is a feed-forward architecture with fully connected layers. Connections between units are characterized by adjustable parameters called weights. This

refers to the strength of a connection between two nodes [23]. Each neuron computes a function of the sum of the weighted inputs. This function is also called *activation function*.

In this work, we used two different MLP-network architectures. Both are characterized by i) an hidden layer of neurons with the hyperbolic tangent activation function f and ii) an output layer with a linear activation function F. The functional model is given by:

$$\hat{y}_i(w, W) = F_i(\sum_{j=1}^{q} W_{ij} h_j + W_{i0}) = F_i(\sum_{j=0}^{q} W_{ij} f_j(\sum_{l=1}^{m} w_{jl} u_l + w_{j0}) + W_{i0}) \quad (1)$$

Weights are specified by the matrices $W = [W_{ij}]$ and $w = [w_{jl}]$; where W_{ij} scales the connection between the hidden unit j and the output unit i and w_{jl} instead the connection between the hidden unit j and the input unit l. The corresponding biases are W_{i0} and w_{j0}. These weights are vectorized in a vector θ. The input units are represented by the vector $u(t)$ while the vector h represents the hidden neuron outputs. The outputs of the network, $\hat{y}_i$, are estimated by Equation 1. The parameters are determined during the *training process*, which requires a *training set* Z^N, composed of a set of inputs, $u(t)$, and corresponding desired outputs, $y(t)$, specified by:

$$Z^N = [u(t), y(t)], \ t = 1, ..., N \quad (2)$$

The training phase allows to determine a mapping from the set of training data to the set of possible weights:

$$Z^N \rightarrow \hat{\theta} \quad (3)$$

the network can predict $\hat{y}(t)$ that can be compared to the true output $y(t)$. The *prediction error approach* is instead based on the introduction of a measure of closeness in terms of a mean square error criterion, as specified by:

$$V_N(\theta, Z^N) = \frac{1}{2N} \sum_{t=1}^{N} [y(t) - \hat{y}(t|\theta)]^T [y(t) - \hat{y}(t|\theta)] \quad (4)$$

Weights are then found as:

$$\hat{\theta} = arg_\theta min V_N(\theta, Z^N) \quad (5)$$

by some kind of iterative minimization scheme:

$$\theta^{i+1} = \theta^i + \mu^i + f^i \quad (6)$$

where θ^i specifies the current iteration, f^i the search direction and μ^i the step size.

2.2 System Identification

This section details the adopted methodology to use an ANN to predict solar radiation in short-term. Based on the methodology presented in [33] and as widely detailed in [2], the procedure to identify a dynamical system consists of four steps: i) *Experiment*, ii) *Model Structure Selection*, iii) *Model Estimation* and iv) *Model Validation* (see Fig. 1).

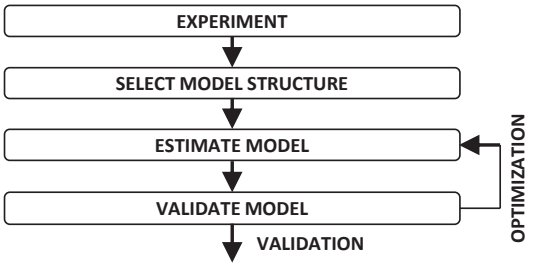


Fig. 1. System identification procedure [2]

The *Experiment* represents the problem analysis, data sampling and collection phase. This phase is the keystone of the whole process. Some of the main issues in this stage consist on: i) choosing the sampling frequency, ii) designing suitable input signals, and iii) pre-processing the dataset. Data pre-processing may include some non-linearity tests and disturbances removal. If the dataset is well organized, all the next steps in Fig. 1 will have less chance to fail. Generally, a big and relevant amount of data is needed to train and validate a neural network. Moreover, an higher number of data allows better forecasting performance [42]. However, it is necessary to avoid overfitting. This means to avoid unknowingly extract some of the residual variation (i.e. the noise) as if that variation represented underlying model structure. [51]. Then, the collected data must be divided into two datasets: training-set and validation-set [19]. Both datasets are used in training and validation phases of the neural network that are *Estimate Model* and *Validate Model* in Fig. 1, respectively. The *Model Structure Selection* phase identifies the correct architecture model and the number of regressors [33]. Generally, this selection is more difficult in nonlinear cases than in linear [9]. In time-series, the regressors represent the previous samplings with respect to the predicted ones [30]. Once the best network model and the appropriate number of regressors are identified, in the *Model Estimation* phase, the network is first implemented and then trained. In time-series scenario, training a neural network is needed to provide: i) the vector containing desired output data; ii) the number of regressors to define the prediction; iii) the vector containing the weights of both input-to-hidden and hidden-to-output layers and lastly iv) the data structure containing the parameters associated with the selected training algorithm. The training process produces a training error, which represents the network

performance index [42]. The *Model Validation* allows validating the trained network in order to evaluate its capabilities [29]. In time-series predictions, the most common validation method consists of analyzing the residuals (i.e. prediction errors) by cross-validating the validation-set [39]. This analysis provides the *test error* [42], that is an index considered as a generalization of the error estimation. This index should not be too high compared to training error, if this happens the network could over-fit the training-set. This means that the selected model structure contains too many weights. In this case, it is required to return in the *Estimate Model* step to change and redefine some structural parameters by optimizing the whole architecture. For this purpose, the superfluous weights must be pruned according to the Optimal Brain Surgeon, which represents one of the most important optimization strategies [17]. Consequently, once new weights are given, the network architecture must be re-validated.

3 Neural networks for short-term GHI forecast

The purpose of this work consists of forecasting short-term Global Horizontal Solar Irradiance (GHI) for photovoltaic energy predictions. To deal with these time-series data, we adopted, and then compared, two ANNs: i) Nonlinear Autoregressive neural network (NAR) and ii) Nonlinear Autoregressive Moving Average neural network (NARMA). NAR belongs to the family of Nonlinear Autoregressive Exogenous Model (NARX) [41]. It is generally considered as one of the best tools for time-series analysis and does not suffer of stability problems [43]. This is due to its nonlinear autoregressive model which has exogenous inputs. This neural network model bases its prediction on i) a variable range of past values and also on ii) the current and past values of the driving exogenous inputs of the time-series in analysis. However, this process produces a prediction error that is the knowledge of the past. Indeed, the presence of this error as result of prediction means that the future values of the time-series cannot be predicted exactly. This family of ANNs is characterized by the following equation:

$$y_t = F(y_{t-1}, y_{t-2}, y_{t-3}, ..., u_t, u_{t-1}, u_{t-2}, u_{t-3}, ...) + \varepsilon_t \tag{7}$$

where y_t is the variable of interest; while u_t represents the externally determined variable at time t in Equation 7. Information about u_t and previous values of u and y helps on predicting y_t with a prediction error ε_t.
On the other hand, NARMA belongs to the family of Nonlinear Autoregressive Moving Average Exogenous Model (NARMAX) [11]. It represents a generalization of the NAR model. However, this model realizes a feed-forward network where a predictor will have a feedback when the regressors are selected. This family is characterized by the following equation:

$$y_t = F(y_{t-1}, y_{t-2}, y_{t-3}, ..., u_t, u_{t-1}, u_{t-2}, u_{t-3}, ...) + C(q^{-1}) + \varepsilon_t \tag{8}$$

where y_t and u_t are the variable of interest and the externally determined variable at time t, respectively; ε_t is the prediction error; C is a polynomial in the backward shift operator expressed as:

$$C(q^{-1}) = 1 + c_1 q^{-1} + ... + c_{nc} q^{-nc} \tag{9}$$

Consequently, the past prediction errors depend on the model output and they are able to establish a feedback.

Comparing the two proposed models, the major difference is that NARMA is a *Recurrent Neural Network* [28], while NAR is not. Thus, NAR has a predictor without feedback while NARMA has feedback through the choice of regressors. Hence, future network inputs will depend on present and past network outputs. This might lead to instability of the ANN itself and it can be very difficult to determine whether or how the predictor is stable. To avoid instability, NARMA architecture uses a linear MA-filter to filter past residuals. This is a Low Pass FIR (Finite Impulse Response) filter, commonly used for smoothing an array of sampled data/signal. It takes a set of inputs at the time, it computes the average of those samples and produces a single output [27].

To implement both NAR and NARMA, we exploit a dataset composed of four years of real GHI values (from 2010 to 2013). These values are 15 minute time sampled by the weather station in our University Campus. Differently to the dataset that we used in our previous work [2], this is more complete and much bigger because it also includes the GHI values in the time period between 6 p.m. and 8 a.m., i.e. all evening and night values. In our previous work, we ignored these values because they are less relevant in the GHI prediction procedure leading to a simpler ANN architecture and avoiding over-fitting. However, in this work we demonstrate that if ANN ignores the whole daily period, that is the succession of night and day, the values of GHI predicted in the early hours of the morning are significantly incorrect (i.e. oversized or undersized). A further difference is the use of data to initialize both training set and validation-set. In our previous work, we split the dataset symmetrically according to [33]. Whilst in this paper, we divided the dataset asymmetrically into three years for the training-set (2010-2012) and one year for the validation-set (2013), according to the more recent approach described in [51]. This allows an even more accurate training phase.

Progressively, we started analyzing the number of past signals used as regressors for the prediction. Specifically, we used *Lipschiz* [38] to determinate the *lag-space*. This methodology allows identifying the order of Input-Output Models for Nonlinear Dynamic Systems. Given corresponding input and output sequences, it calculates a matrix of indices that can be helpful for determining a proper lag-space structure. However, as detailed in [20], this methodology is not always effective but it represents a good starting point to define the more suitable number of regressors, which will characterize the future neural networks architectures. Consequently, we started the design of both our ANNs considering a number of regressors in the range between 1 and 10. This arbitrary choice derives from our previous work [2] that exploits 10 regressors. On the other hand, ANNs proposed in this work aim at improving previous performances. Moreover as previously described, we modified the pre-process of the input dataset, this

allows designing an ANN with no more than 10 regressors. Fig. 2 details the result of the applied lag-space investigation methodology.

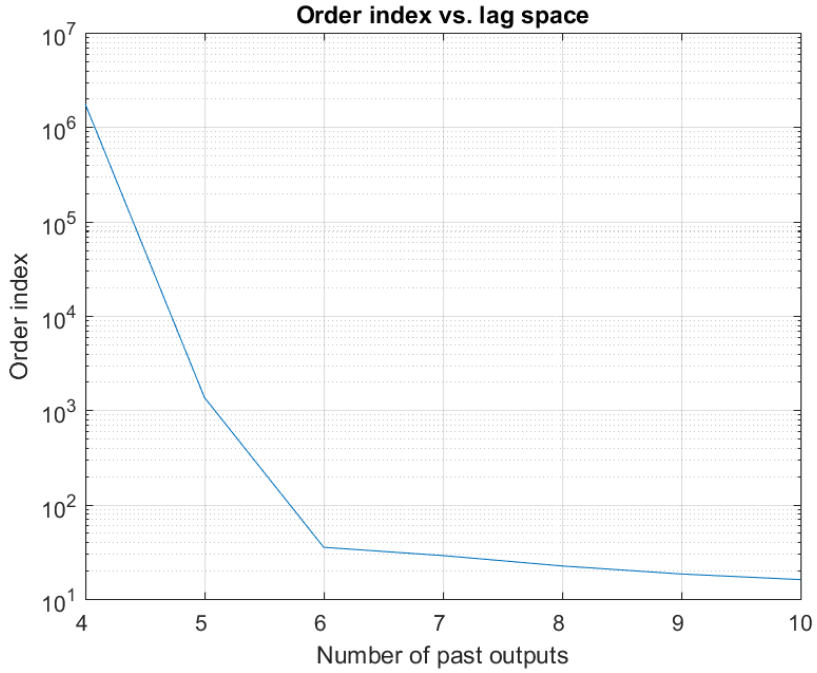


Fig. 2. Evaluation of Order Index criterion for different lag-space

Fig. 2 suggests that good performance can be achieved with 6 regressors (i.e. 6 previous values for y and u in Equation 7, respectively). All previous values are not computationally advantageous. Even, between 1 and 4, the result diverges to infinity and therefore they are not displayed in the plot. On the other hand, all values above 6, even if advantageous, would risk transforming ANN architecture into a more complex and less performing structure. Thus, the best configuration in the computation/performance ratio is achieved with 6 regressors (see the knee-point of the plot in Fig. 2). Differently to what we did in [2], in this work we adopted also a design space exploration approach [10]. As a result, we decide to validate (or refute) the results given by Lipschitz. For this purpose, we implemented all the possible network combinations (of both NAR and NARMA models) from 1 to 10 regressors. This allows to evaluate and compare all the obtained architectures' performance and then find the best solutions for both NAR and NARMA. Fig. 3 shows the NSSE error trends of the two ANNs based on the number of regressors for each implemented architecture. NSSE error is a network performance index. The lower NSSE the better ANN's performance.

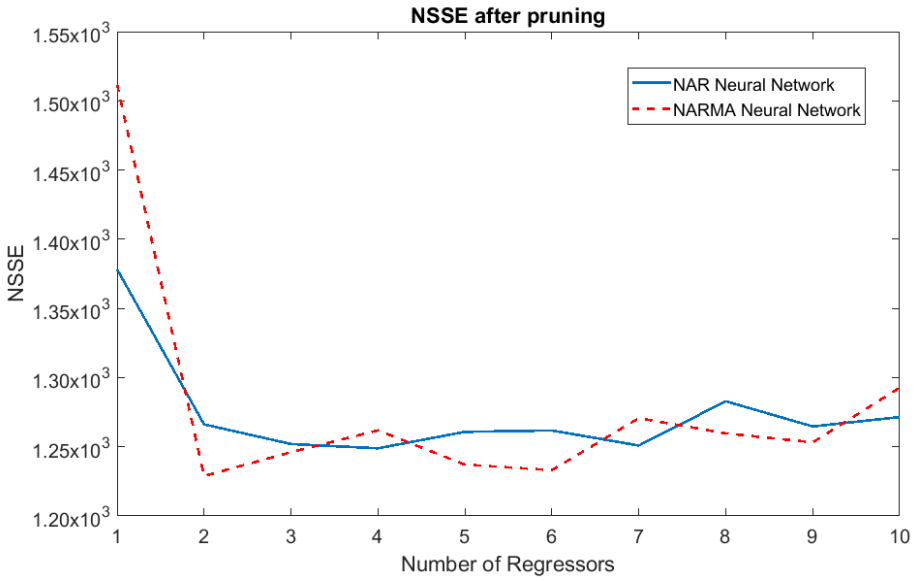


Fig. 3. Evaluation of NSSE after pruning with regard to number of regressors

As shown in Fig. 3, NAR and NARMA give the best performance with 4 and 2 regressors, respectively. This also represents the best compromise between ANN's computation and performance. It is worth noting that these NSSE results improve the indication given by Lipschitz methodology that suggested 6 as the best number of regressors (see Fig. 2).

Once we found the optimal regressors for both NAR and NARMA, we implemented the two final ANNs starting from two fully connected architectures with one hidden layer of 30 hyperbolic tangent units. This large number of units could be redundant, but it is justified by the pruning technique [44], which is used in the next phases to optimize the network architecture themselves.

Before training, weights of both ANNs are initialized randomly. This allows also to initialize i) weights, ii) their decay threshold and iii) the maximum number of iterations. However, these parameters are overestimated during the very first training iteration. Then, we proceeded with training phase for both NAR and NARMA networks. Training is a minimizing technique to compute the best weights. For both architectures, we used the *Levenberg-Marquardt* algorithm, which interpolates between the Gauss-Newton algorithm and the method of gradient descent using a *trust region* approach [33]. Progressively, we used the methodology illustrated in [32] for validating both ANNs. This methodology performs a set of tests including autocorrelation function of residuals and cross-correlation function between controls and residuals to validate system outputs. The result of this process gives the NSSE error. By definition, this error should not be too large compared to training error. If NSSE is greater than the training

error, the predicted results are over-fitting the training-set. Table 1 illustrates the obtained results for both NAR and NARMA.

Table 1. NSSE comparison after the first and final validation

Neural Network	NSSE after first validation	NSSE after final validation
NAR	$1.25 \times 10^{+3}$	$1.24 \times 10^{+3}$
NARMA	$1.23 \times 10^{+3}$	$1.22 \times 10^{+3}$

A shown in Table 1, the validation process yields these indexes as detailed in the column *NSSE after first validation*. The NSSE is equal to $1.25 \times 10^{+3}$ and $1.23 \times 10^{+3}$ for NAR and NARMA, respectively. These indexes will have to be compared with those obtained after the optimization of the architectures. Then, we proceeded to the optimization phase of both networks. Our purpose was to remove excess weights and obtain a smaller error than the one given during the first validation. To achieve this, we adopted the *Optimal Brain Surgeon* (OBS) [18], which is a technique to prune superfluous weights. OBS computes the Hessian matrix weights iteratively, which leads to a more exact approximation of the error function. The inverse Hessian is calculated by means of recursion. This method allows finding the smallest saliency S_i as follows:

$$S_i = \frac{w_i^2}{2[H^{-1}]_{i,i}} \tag{10}$$

where $[H^{-1}]_{i,i}$ is the $(i,i)th$ element of the inverse Hessian matrix and w_i represent the ith element of the vector θ containing the network weights. The saliency identifies the quality of the connection between the various network units. This methodology allows verifying the state of the saliency iteratively. If the saliency S_i is much smaller than the mean-square error, then some synaptic weights are deleted and the remaining ones are updated. The computation stops when no more weights can be removed from the network without a large increase of the mean-square error. Once the new weights are given, we re-validated both resulting pruned NAR and NARMA.

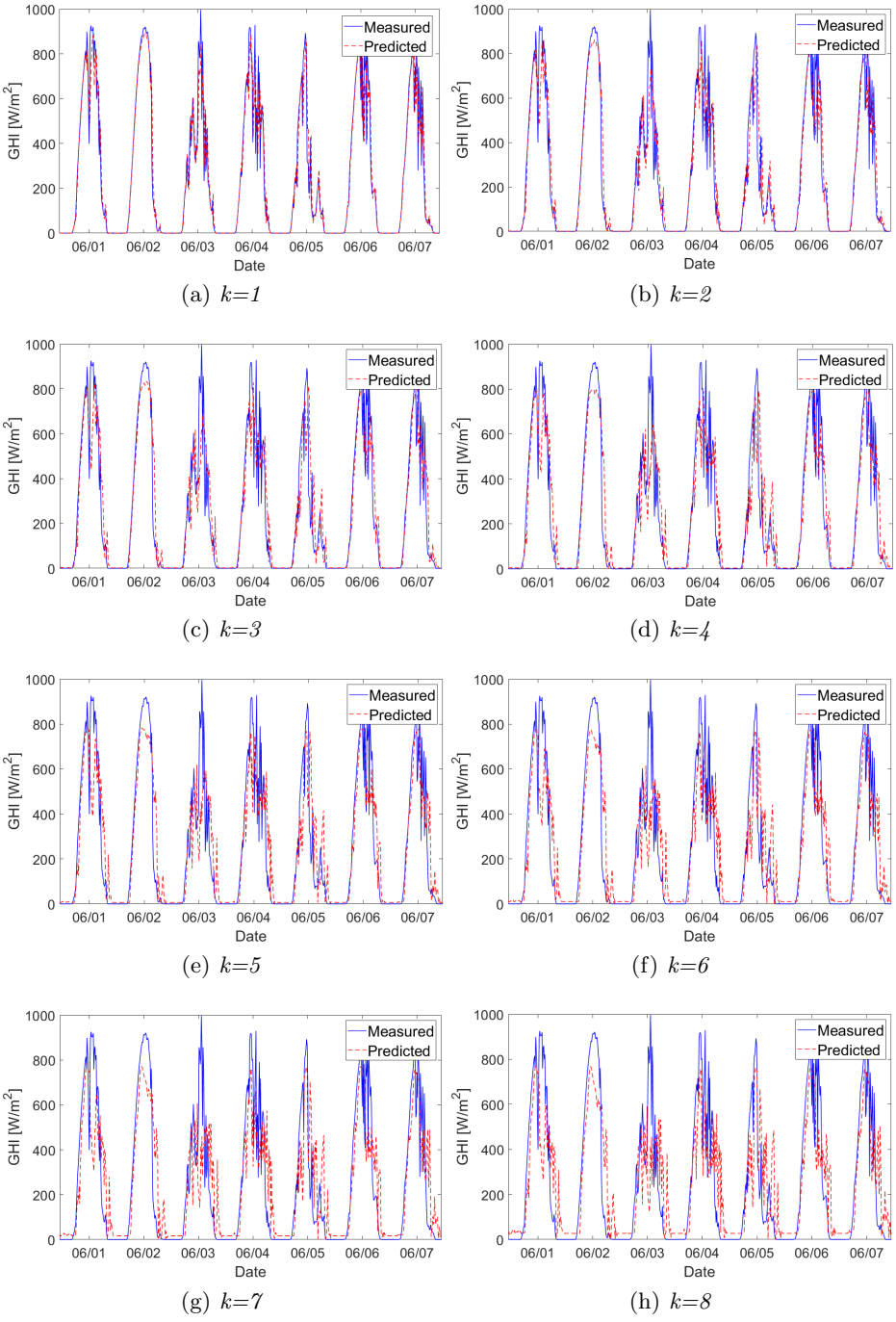
Through the same methodology used in the first validation phase, we proceeded to the final networks validation using the new weights. The resulting NSSE error indexes for both ANNs are illustrated in the column *NSSE after final validation* in Table 1. NSSE error indexes are $1.24 \times 10^{+3}$ and $1.22 \times 10^{+3}$ for NAR and NARMA, respectively. In both cases, the new NSSE values are lower than the ones given after the first network validation. Thus, the optimization for both ANNs succeeded. The resulting NAR with 4 regressors and NARMA with 2 regressors are trained and validated. Hence, they are ready to forecast GHI values in short-term time-periods and their results will be discussed in next Section 4.

4 Results on GHI forecast

The purpose of our work is to predict GHI in short time windows, i.e. 15 minutes. This is the minimum time interval on which many services for smart grid management work (e.g. Demand/Response [40]). However, we moved further predicting also GHI up to next two hours (again with 15 min time intervals). As described in Section 3, we implemented two Non-linear Autoregressive Neural Networks, i) NAR with 4 regressors and ii) NARMA with 2 regressors, that exploit the dataset described in Section 3. In this section, we present the obtained results and we also compare and discuss the two different architectures.

To evaluate the performance of our networks, we compare the results of our predictions with real measured values. To achieve this, we used a set of indicators introduced by Gueymard et al. [15] and briefly presented in the following. *Root Mean Square Difference (RMSD)* represents the standard deviation of differences between predicted and observed values. *Mean Absolute Difference (MAD)* represents a measure of statistical dispersion obtained by the average absolute difference of two independent values drawn from a probability distribution. *Mean Bias Difference (MBD)* measures the average squares of errors between predicted and measured values. *Coefficient of determination (r^2)* represents the proportion between the variance and the predicted variable. All these values are expressed in percentage. Finally, we also considered two other indicators to evaluate the overall network performance: *Willmotts Index of Agreement (WIA)* and *Legatess Coefficient of Efficiency (LCE)*. WIA represents the standardized measure of the degree of model prediction error [50]. LCE is the ratio between the mean square error and the variance in the observed data [24].

Fig. 4 and Fig. 5 show the results of predictions given by proposed NAR and NARMA compared with real measured values sampled by weather station (dashed and continuous lines, respectively). These results include predictions with eight different time-steps, from $k = 1$ (i.e. next 15 min) to $k = 8$ (i.e next 120 min). Both cases refer to the first seven days of June 2013. Prediction trends of both architectures are very similar. Indeed, they follow with a good accuracy the real meteorological trends: i) clear sky, ii) cloudy and iii) rainy conditions, especially for $1 \leq k \leq 3$. Instead for $k > 3$, the prediction accuracy decreases. These aspects are better highlighted by Table 2 that reports the results of GHI predictions in terms of performance indicators considering the whole 2013, which is our validation-set for both architecture. These indexes highlight that the prediction performance worsens by increasing the predictive k-steps. Indeed, GHI predictions for high values of k has a higher error compared with real measurements. In both cases, the analysis of indexes highlights that the best GHI predictions are given with smaller time intervals. For example, MAD reveals that the forecast error grows as the prediction step k increases. Indeed, for $k = 1$, the error is about 12.96% and 12.81% for NAR and NARMA, respectively. Whilst for $k = 8$, the error exceeds the 55%. Also, $RMSD$ has a similar trend. A good performance of r^2 is given when its values are closer to 1. In both cases, this happens for lower k values. For $k = 1$ and $k = 2$, r^2 is over 0.92 for both ANNs. For $k \geq 4$, it decreases down to about 0.70. This trend is also confirmed



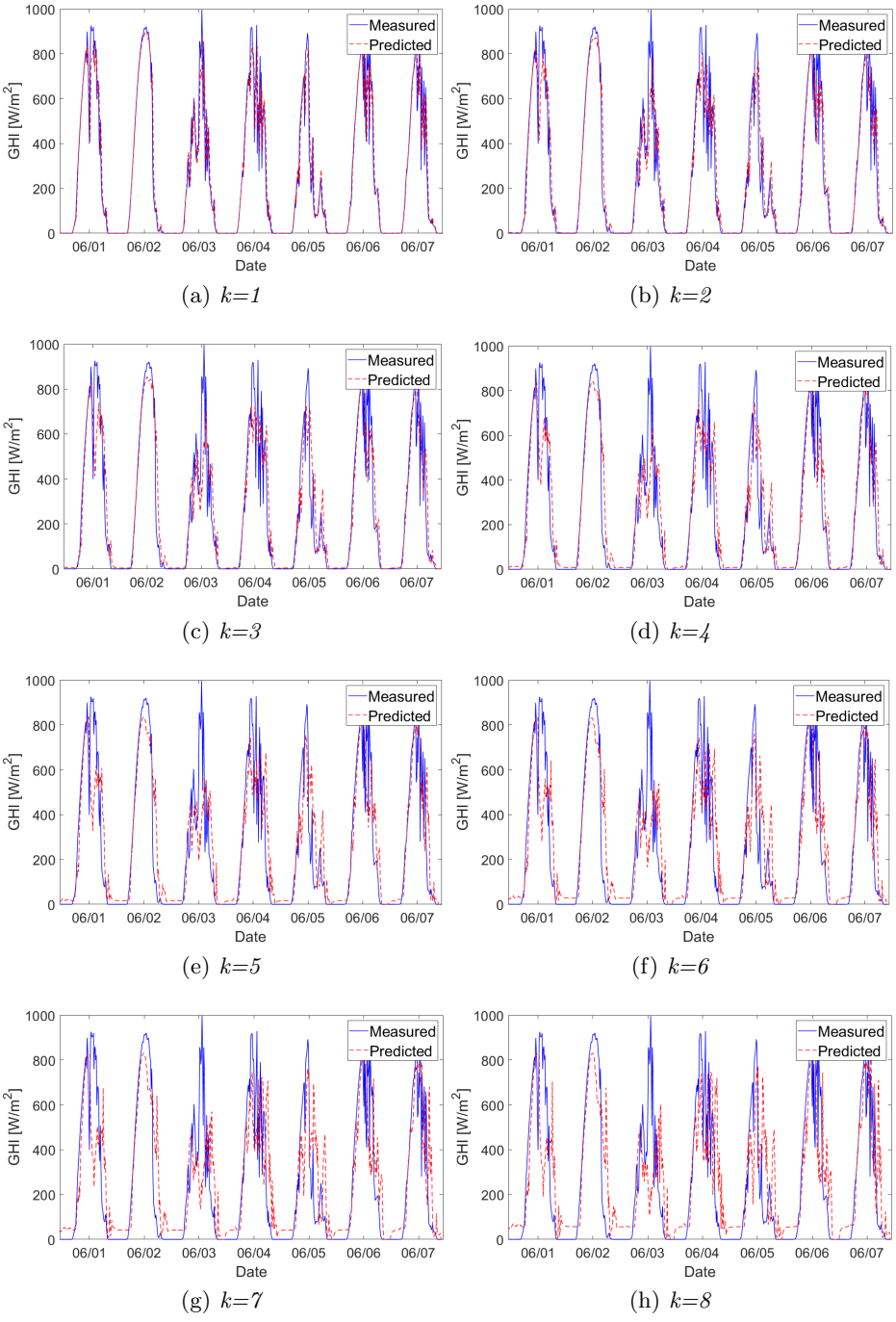
(a) *k=1* (b) *k=2*

(c) *k=3* (d) *k=4*

(e) *k=5* (f) *k=6*

(g) *k=7* (h) *k=8*

Fig. 4. NAR GHI prediction for $1 \leq k \leq 8$ (June 2013)



(a) *k=1* (b) *k=2*

(c) *k=3* (d) *k=4*

(e) *k=5* (f) *k=6*

(g) *k=7* (h) *k=8*

Fig. 5. NARMA GHI prediction for $1 \leq k \leq 8$ (June 2013)

by both *LCE* and *WIA* that highlight a decreasing of the overall performance on high prediction steps. Under these circumstances, the performance indexes for $1 \leq k \leq 3$ are suitable to perform Photovoltaic energy estimations and simulation results will be discussed in the next Section 5. With this configuration, the maximum error rate for GHI prediction (expressed in *MAD*) is less than 25%.

Table 2. Performance Indicators for GHI predictions

		NAR Neural Network						NARMA Neural Network					
Pred. Steps	Time [min]	MAD [%]	MDB [%]	r^2	RMSD [%]	LCE	WIA	MAD [%]	MDB [%]	r^2	RMSD [%]	LCE	WIA
k=1	15	12.94	0.16	0.95	35.31	0.89	0.99	12.80	0.36	0.95	35.03	0.90	0.99
k=2	30	19.47	0.81	0.92	46.80	0.84	0.98	19.15	1.06	0.92	46.07	0.84	0.98
k=3	45	24.87	1.80	0.89	54.41	0.80	0.97	24.67	2.23	0.89	53.65	0.80	0.97
k=4	60	30.16	3.06	0.86	61.05	0.76	0.96	30.23	4.09	0.86	60.24	0.76	0.96
k=5	75	35.67	4.77	0.83	67.29	0.71	0.95	36.35	6.86	0.83	66.82	0.71	0.95
k=6	105	41.78	7.23	0.79	73.93	0.66	0.94	43.30	10.46	0.79	74.16	0.65	0.94
k=7	115	48.90	10.65	0.75	80.94	0.60	0.92	50.61	14.57	0.74	81.88	0.60	0.92
k=8	120	56.90	15.23	0.70	88.51	0.54	0.90	58.18	19.01	0.69	89.89	0.53	0.90

To train and validate the proposed ANNs, we run our simulations in a server equipped with a CPU 2x Intel Xeon E5-2680 v3 2.50 GHz and 128 Gb of RAM. Table 3 reports the execution time for both ANNs considering the three main phases: i) *ANN initialization before Pruning*, ii) *Pruning* and iii) *ANN initialization after Pruning*.

ANN initialization before Pruning refers to the computational time needed to initialize ANNs with random values for the first training and validation. It includes all the steps needed before carrying out the network pruning. As shown in Table 3, NAR with 4 regressors needs about 1 min. Whilst, NARMA with 2 regressors needs about 2:30 min because its overall architecture is more complex with respect to NAR; hence, it needs more computational resources. This is clearly highlighted during the *Pruning* in Table 3, which refers to computational time to evaluate and eliminate unnecessary weights in order to optimize ANNs. This procedure takes around 1 hour for NAR and about 1:48 hour for NARMA. Thus, the optimization process is almost doubled for NARMA with respect to NAR. Finally, *ANN initialization after Pruning* is the time needed to train and validate the optimized ANNs. As highlighted in Table 3, it dropped to 1:22 min for NARMA with respect to the previous *ANN initialization before Pruning*. Whilst, it is almost constant for NAR.

Once both ANNs are pruned, the computation time to provide GHI forecasts varies between 14 and 21 seconds for $1 \leq k \leq 8$. This enables possible future applications where these ANNs are trained, validated and pruned on servers or cluster systems, since these phases need more computational resources. Then,

Table 3. Computation time for both NAR and NARMA

			NAR	NARMA
Execution Time [hh:mm:ss]	ANN initialization before Pruning		00:01:13	00:02:24
	Pruning		01:04:12	01:47:58
	ANN initialization after Pruning		00:01:07	00:01:22
	Total		01:06:32	01:51:44
	Prediction step	k = 1	00:00:14	00:00:14
		k = 2	00:00:13	00:00:14
		k = 3	00:00:14	00:00:14
		k = 4	00:00:16	00:00:18
		k = 5	00:00:18	00:00:16
		k = 6	00:00:19	00:00:19
		k = 7	00:00:19	00:00:20
		k = 8	00:00:20	00:00:21

the optimized ANNs can be deployed on embedded devices to provide GHI forecast. In a smart grid scenario, examples of application that can benefit from this forecast are: i) energy dispatching and load balancing [48], ii) battery management system [36], iii) Demand/Response services [40] and iv) vehicle-to-grid applications [37], [52].

5 PV energy estimation

As already discussed in Section 4, the proposed ANNs forecast GHI in short-term time windows with a good accuracy. This allows estimating in advance energy produced by PV systems. To achieve this, we exploited the PV energy simulator (PVsim) presented in [6] that takes as input the GHI forecast resulting by both NAR and NARMA. The combination of both ANNs and PVsim unlocks development of novel services and control policies for a better management of future smart grids [40] that can also be tested and validated exploiting the methodology in [4].

PVsim is a GIS software infrastructure that simulates PV production in real-sky conditions. The inputs for these simulations are i) a *Digital Surface Model* (DSM) and ii) GHI trends. DSM is a digital elevation model that represents terrain elevation including all objects on it (i.e. buildings). It is used by PVsim to identify rooftops and to simulate the evolution of shadows in clear-sky conditions during the day. Then, this is combined with GHI trends to simulate solar incident radiation and, consequently, PV production in real-sky conditions with a time-resolution of 15 minutes. In a default configuration, PVsim retrieves real GHI trends from a weather station in our University Campus. To forecast PV energy

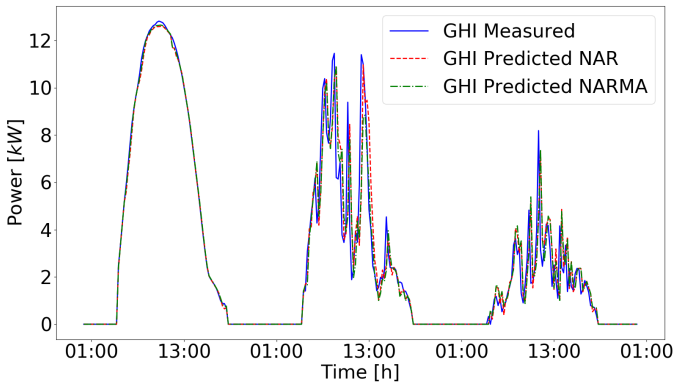
production in short-term time intervals, we interpose our ANNs between weather station's data-source and PVsim. So that, both ANNs get the last real GHI measurements from the weather station and provide the resulting GHI forecast to PVsim.

As mentioned in Section 4, results of our ANNs for $1 \leq k \leq 3$ on GHI forecast are suitable to perform PV energy estimations. Hereafter, we present results obtained for these three time intervals, i.e. next 15, 30 and 45 minutes. To evaluate the error rate, we compared PVsim results of GHI forecast trends given by NAR and NARMA with those of real GHI trends retrieved by weather station. Fig. 6 shows PVsim results for three significant days in June 2013 with different meteorological conditions: i) sunny, ii) cloudy and iii) rainy. Blue continuous-line represents simulations given by real GHI trends, red dashed-line given by NAR GHI trends and green dashed-dotted-line given by NARMA GHI trends. As shown in Fig. 6, best performance is achieved when PVsim gets as input results of GHI trends from both ANNs with $k = 1$. This is also confirmed by performance indicators reported in Table 4 that considers the whole 2013. Indeed, the accuracy of PV energy estimations decreases by increasing the prediction step k. Regarding PVsim simulations preformed with NAR GHI trends, *MAD* increases from 10.31% to 19.22% for $k = 1$ and $k = 3$, respectively. Also *RMSD* has a similar trend, increasing from 27.96% to 44.65%. *MDB* varies from -0.61 to -2.65. r^2 for $k = 1$ is equal to 0.97. Whilst, the error increases with an $r^2 = 0.92$ for $k = 3$. Finally, *LCE* varies from 0.92 to 0.84 and *WIA* decreases from 0.99 to 0.97. Similar trends are achieved by PVsim simulations preformed with NARMA GHI trends. *MAD* increases from 10.11% for $k = 1$ to 18.47% for $k = 3$. *MDB* is -0.17 for $k = 1$, -0.74 for $k = 2$ and -1.55 for $k = 3$. r^2 varies from 0.97 to 0.92. *RMSD* rises from 27.86% to 43.97%. Finally, *LCE* varies from 0.91 to 0.85 and WIA decreases from 0.99 to 0.98.

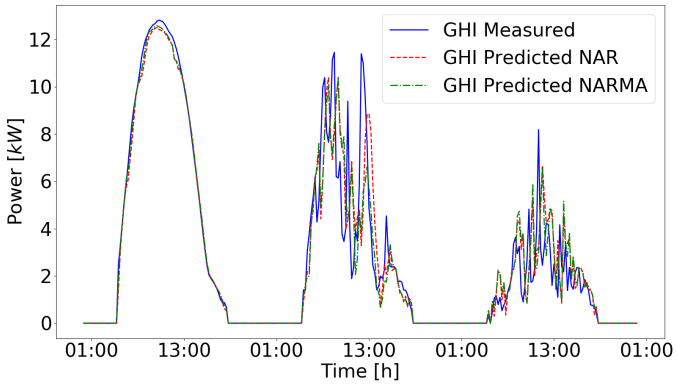
Table 4. Performance Indicator for PV simulation with NAR and NARMA

		NAR Neural Network						NARMA Neural Network					
Pred. Steps	Time [min]	MAD [%]	MDB [%]	r2	RMSD [%]	LCE	WIA	MAD [%]	MDB [%]	r2	RMSD [%]	LCE	WIA
k=1	15	10.31	-0.61	0.97	27.96	0.92	0.99	10.11	-0.17	0.97	27.86	0.91	0.99
k=2	30	15.45	-1.58	0.94	38.15	0.87	0.98	14.93	-0.74	0.94	37.66	0.88	0.98
k=3	45	19.22	-2.65	0.92	44.65	0.84	0.97	18.47	-1.55	0.92	43.97	0.85	0.98

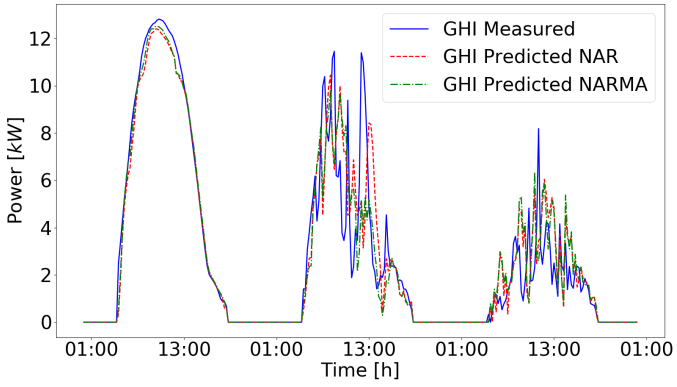
A comparison of these performance indicators highlights that NARMA GHI trends give slightly better PV energy estimations than NAR GHI trends. We can assert that both ANNs are able to simulate GHI trends in short-term periods with a good accuracy. This is also confirmed when we use ANNs' results to estimate PV energy production.



(a) *k=1*



(b) *k=2*



(c) *k=3*

Fig. 6. Simulations of PV energy production given by real NAR and NARMA GHI trends for $1 \leq k \leq 3$ (June 2013)

6 Conclusions

In this paper, we presented a methodology to implement two artificial neural networks for forecasting solar radiation in short-term time periods. In a smart grid scenario, this forecast is needed to estimate in advance the energy production of renewable energy sources enabling novel control strategies and services for grid management, such as Demand/Response policies [40].

The proposed ANNs implement two non-linear autoregressive neural networks, NAR and NARMA respectively, that exploit time-series data. We also analyzed their accuracy by comparing the obtained results with real values of solar radiation sampled by a real weather station. This analysis highlighted an overall good performance especially for a time horizon from future 15 to 120 minutes. We discussed strengths and weaknesses of both architectures, from both neural and computational viewpoints. Then, the results of these ANNs have been given as input to a Photovoltaic simulator (PVsim) [6] to estimate the energy production of Photovoltaic systems. The accuracy of these results is also acceptable especially for time horizon from future 15 to 45 minutes.

As future work, we will apply these ANNs together with PVsim to test services for smart grid management in a distributed test-bed environment, as depicted in [5].

Acknowledgements

Computational resources were provided by HPC@POLITO, a project of Academic Computing within the Department of Control and Computer Engineering at the Politecnico di Torino (http://www.hpc.polito.it)

References

1. Aghaei, J., Alizadeh, M.I.: Demand response in smart electricity grids equipped with renewable energy sources: A review. Renewable and Sustainable Energy Reviews **18**, 64–72 (2013)
2. Aliberti, A., Bottaccioli, L., Cirrincione, G., Macii, E., Acquaviva, A., Patti, E.: Forecasting short-term solar radiation for photovoltaic energy predictions. In: Proceedings of the 7th International Conference on Smart Cities and Green ICT Systems - Volume 1: SMARTGREENS,. pp. 44–53. INSTICC, SciTePress (2018). https://doi.org/10.5220/0006683600440053
3. Badescu, V.: Modeling solar radiation at the earth's surface. Springer (2014)
4. Bottaccioli, L., Estebsari, A., Patti, E., Pons, E., Acquaviva, A.: A novel integrated real-time simulation platform for assessing photovoltaic penetration impacts in smart grids. Energy Procedia **111**, 780–789 (2017)
5. Bottaccioli, L., Estebsari, A., Pons, E., Bompard, E., Macii, E., Patti, E., Acquaviva, A.: A flexible distributed infrastructure for real-time cosimulations in smart grids. IEEE Transactions on Industrial Informatics **13**(6), 3265–3274 (2017)
6. Bottaccioli, L., Patti, E., Macii, E., Acquaviva, A.: Gis-based software infrastructure to model pv generation in fine-grained spatio-temporal domain. IEEE Systems Journal (2017)

7. Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time series analysis: forecasting and control. John Wiley & Sons (2015)
8. Brockwell, P.J., Davis, R.A.: Introduction to time series and forecasting. springer (2016)
9. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Computers & Electrical Engineering **40**(1), 16–28 (2014)
10. Cohn, D.A.: Neural network exploration using optimal experiment design. In: Advances in neural information processing systems. pp. 679–686 (1994)
11. Connor, J., Atlas, L.E., Martin, D.R.: Recurrent networks and narma modeling. In: Advances in Neural Information Processing Systems. pp. 301–308 (1992)
12. Demuth, H.B., Beale, M.H., De Jess, O., Hagan, M.T.: Neural network design. Martin Hagan (2014)
13. Dickinson, E.: Solar energy technology handbook. CRC Press (2018)
14. Expósito, A.G., Conejo, A.J., Canizares, C.: Electric energy systems: analysis and operation. CRC press (2016)
15. Gueymard, C.A.: A review of validation methodologies and statistical performance indicators for modeled solar radiation data: Towards a better bankability of solar projects. Renewable and Sustainable Energy Reviews **39**, 1024–1034 (2014)
16. Hamilton, J.D.: Time series analysis, vol. 2. Princeton university press Princeton (1994)
17. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: Advances in Neural Information Processing Systems. pp. 1135–1143 (2015)
18. Hansen, L.K., et al.: Controlled growth of cascade correlation nets. In: ICANN94, pp. 797–800. Springer (1994)
19. Haykin, S., Network, N.: A comprehensive foundation. Neural networks **2**(2004), 41 (2004)
20. He, X., Asada, H.: A new method for identifying orders of input-output models for nonlinear dynamic systems. In: American Control Conference, 1993. pp. 2520–2523. IEEE (1993)
21. Hosenuzzaman, M., Rahim, N., Selvaraj, J., Hasanuzzaman, M., Malek, A., Nahar, A.: Global prospects, progress, policies, and environmental impact of solar photovoltaic power generation. Renewable and Sustainable Energy Reviews **41**, 284–297 (2015)
22. Kaplanis, S., Kaplani, E.: Stochastic prediction of hourly global solar radiation profiles (2016)
23. Kubat, M.: Artificial neural networks. In: An Introduction to Machine Learning, pp. 91–111. Springer (2017)
24. Legates, D.R., McCabe, G.J.: A refined index of model performance: a rejoinder. International Journal of Climatology **33**(4), 1053–1056 (2013)
25. Ljung, L.: System identification. In: Signal analysis and prediction, pp. 163–173. Springer (1998)
26. Madanchi, A., Absalan, M., Lohmann, G., Anvari, M., Tabar, M.R.R.: Strong short-term non-linearity of solar irradiance fluctuations. Solar Energy **144**, 1–9 (2017)
27. Makridakis, S., Wheelwright, S.C.: Adaptive filtering: An integrated autoregressive/moving average filter for time series forecasting. Journal of the Operational Research Society **28**(2), 425–437 (1977)
28. Mandic, D.P., Chambers, J.A., et al.: Recurrent neural networks for prediction: learning algorithms, architectures and stability. Wiley Online Library (2001)

29. Miller, G.F., Todd, P.M., Hegde, S.U.: Designing neural networks using genetic algorithms. In: ICGA. vol. 89, pp. 379–384 (1989)
30. Montgomery, D.C., Jennings, C.L., Kulahci, M.: Introduction to time series analysis and forecasting. John Wiley & Sons (2015)
31. Nazaripouya, H., Wang, B., Wang, Y., Chu, P., Pota, H., Gadh, R.: Univariate time series prediction of solar power using a hybrid wavelet-arma-narx prediction method. In: Transmission and Distribution Conference and Exposition (T&D), 2016 IEEE/PES. pp. 1–5. IEEE (2016)
32. Norgaard, M., Ravn, O., Poulsen, N.K.l.: Nnsysid-toolbox for system identification with neural networks. Mathematical and computer modelling of dynamical systems **8**(1), 1–20 (2002)
33. Norgaard, P.M., Ravn, O., Poulsen, N.K., Hansen, L.K.: Neural networks for modelling and control of dynamic systems-a practitioner's handbook (2000)
34. Oancea, B., Ciucu, Ş.C.: Time series forecasting using neural networks. arXiv preprint arXiv:1401.1333 (2014)
35. Qazi, A., Fayaz, H., Wadi, A., Raj, R.G., Rahim, N., Khan, W.A.: The artificial neural network for solar radiation prediction and designing solar systems: a systematic literature review. Journal of Cleaner Production **104**, 1–12 (2015)
36. Rahimi-Eichi, H., Ojha, U., Baronti, F., Chow, M.Y.: Battery management system: An overview of its application in the smart grid and electric vehicles. IEEE Industrial Electronics Magazine **7**(2), 4–16 (2013)
37. Rajakaruna, S., Shahnia, F., Ghosh, A.: Plug in electric vehicles in smart grids. Springer (2016)
38. Rajamani, R.: Observers for lipschitz nonlinear systems. IEEE transactions on Automatic Control **43**(3), 397–401 (1998)
39. Refaeilzadeh, P., Tang, L., Liu, H.: Cross-validation. Encyclopedia of database systems pp. 1–7 (2016)
40. Siano, P.: Demand response and smart gridsa survey. Renewable and Sustainable Energy Reviews **30**, 461–478 (2014)
41. Siegelmann, H.T., Horne, B.G., Giles, C.L.: Computational capabilities of recurrent narx neural networks. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **27**(2), 208–215 (1997)
42. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of machine learning research **15**(1), 1929–1958 (2014)
43. Tealab, A., Hefny, H., Badr, A.: Forecasting of nonlinear time series using artificial neural network. Future Computing and Informatics Journal (2017)
44. Thimm, G., Fiesler, E.: Pruning of neural networks. Tech. rep., IDIAP (1997)
45. Vardakas, J.S., Zorba, N., Verikoukis, C.V.: A survey on demand response programs in smart grids: Pricing methods and optimization algorithms. IEEE Communications Surveys & Tutorials **17**(1), 152–178 (2015)
46. Voyant, C., Darras, C., Muselli, M., Paoli, C., Nivet, M.L., Poggi, P.: Bayesian rules and stochastic models for high accuracy prediction of solar radiation. Applied Energy **114**, 218–226 (2014)
47. Voyant, C., Notton, G., Kalogirou, S., Nivet, M.L., Paoli, C., Motte, F., Fouilloy, A.: Machine learning methods for solar radiation forecasting: A review. Renewable Energy **105**, 569–582 (2017)
48. Weckx, S., Driesen, J.: Load balancing with ev chargers and pv inverters in unbalanced distribution grids. IEEE Transactions on Sustainable Energy **6**(2), 635–643 (2015)

49. Weigend, A.S.: Time series prediction: forecasting the future and understanding the past. Routledge (2018)
50. Willmott, C.J., Robeson, S.M., Matsuura, K.: A refined index of model performance. International Journal of Climatology **32**(13), 2088–2094 (2012)
51. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2016)
52. Xing, H., Fu, M., Lin, Z., Mou, Y.: Decentralized optimal scheduling for charging and discharging of plug-in electric vehicles in smart grids. IEEE Transactions on Power Systems **31**(5), 4118–4127 (2016)
53. Yadav, A.K., Chandel, S.: Solar radiation prediction using artificial neural network techniques: A review. Renewable and Sustainable Energy Reviews **33**, 772–781 (2014)